```
from datetime import datetime
from time import sleep
import warnings
from dotenv import dotenv_values
from threading import *
import requests
from dateparser import parse

class Api:
warnings.simplefilter('ignore')
__key = dotenv_values(".env")
__key = __key["key"]
__apiMap = {}
__mainApiMap = {}
__url = "https://newscatcher.p.rapidapi.com/v1/latest_headlines"
__headers = {
"X-RapidAPI-Key": str(__key),
"X-RapidAPI-Host": "newscatcher.p.rapidapi.com"
}
def __newCatcherRunner(self, title):
querystring = {"topic": title, "lang": "en",
"media": "True", "country": "IN"}
respone = requests.request(
"GET", url=self.__url, headers=self.__headers, params=querystring)
respone = respone.json()
retArr = []
for x in respone["articles"]:
newJson = {}
newJson["url"] = x["link"]
newJson["title"] = x["title"]
newJson["img"] = x["media"]
newJson["topic"] = x["topic"]
currTime = parse(x["published_date"])
newJson["date"] = currTime.strftime("%d/%m/%Y")
retArr.append(newJson)
return retArr
def __topHeadlinesFetcher(self):
querystring = {"topic":"news","lang": "en","media": "True", "country": "IN"}
respone = requests.request("GET", url=self.__url, headers=self.__headers, params=querystring)
respone = respone.json()
retArr = []
for x in respone["articles"]:
newJson = {}
newJson["url"] = x["link"]
newJson["title"] = x["title"]
newJson["img"] = x["media"]
newJson["topic"] = x["topic"]
currTime = parse(x["published_date"])
newJson["date"] = currTime.strftime("%d/%m/%Y")
retArr.append(newJson)
self.__apiMap["headline"]=retArr
print("headline fetched at "+str(datetime.now()))
def __newsCatcherApiFetcher(self):
arr = ["sport", "tech", "world", "finance", "politics", "business",
```

```
"economics", "entertainment", "beauty", "travel", "music", "food", "science"]
for x in arr:
self.__apiMap[x] = self.__newCatcherRunner(x)
print("NewsCatcher fetched at "+str(datetime.now()))
def __cricketFetcher(self):
url = "https://cricbuzz-cricket.p.rapidapi.com/news/v1/index"
headers = {
"X-RapidAPI-Key": self.__key,
"X-RapidAPI-Host": "cricbuzz-cricket.p.rapidapi.com"
}
response = requests.request("GET", url, headers=headers)
response = response.json()
response = response["storyList"]
retArr = []
for x in response:
try:
x = x["story"]
newJson = {}
newJson["url"] = f'https://www.cricbuzz.com/cricket-news/{x["id"]}/newsTrakcer'
newJson["title"] = x["hline"]
newJson["image"] = f'https://www.cricbuzz.com/a/img/v1/500x500/i1/c{x["id"]}/abc.jpg'
currTime = datetime.fromtimestamp(int(x["pubTime"])/1e3)
newJson["date"] = currTime.strftime("%d/%m/%Y")
newJson["topic"] = "cricket"
retArr.append(newJson)
except:
pass
self.__apiMap["cricket"] = retArr
print("Cricbuzz fetched at "+str(datetime.now()))
def newsCatcherThreader(self):
while True:
print("NewsCatcher fetching.... at "+str(datetime.now()))
try:
self.__newsCatcherApiFetcher()
self.__mainApiMap = self.__apiMap
except:
print("Error NewsCatcher fetching.... at "+str(datetime.now()))
pass
sleep(30*60)
def topHeadlinesThreader(self):
while True:
print("Headline fetching.... at "+str(datetime.now()))
try:
self.__topHeadlinesFetcher()
self.__mainApiMap = self.__apiMap
except:
print("Error headline fetching.... at "+str(datetime.now()))
pass
sleep(30*60)
def cricbuzzThreader(self):
while True:
print("Cricbuzz fetching.... at "+str(datetime.now()))
try:
self.__cricketFetcher()
self.__mainApiMap = self.__apiMap
```

```
except:
print("Error Cricbuzz fetching.... at "+str(datetime.now()))
pass
sleep(15*60)
def dataGetter(self, topic):
return self.__mainApiMap[str(topic)]
a = Api()
def apiRunner():
t1 = Thread(target=a.topHeadlinesThreader)
t2 = Thread(target=a.newsCatcherThreader)
t3 = Thread(target=a.cricbuzzThreader)
t1.daemon=True
t2.daemon=True
t3.daemon=True
t1.start()
t2.start()
t3.start()
def apiData(topic):
return a.dataGetter(topic)
```